**AMENDMENTS TO THE SPECIFICATION:**

Please amend the specification as follows:

On page 5, please amend the paragraph starting on line 19 as follows:

In one embodiment, the network 10 is organized in a "client-server" configuration, in which one or more computers, shown in FIG. 1 as computers 12(m), operate as servers, and the other computers, shown in FIG. 1 as computers 11(n), operate as clients. In one aspect, one or more of the server computers 12(m) may, as "file servers," include large-capacity mass storage devices which can store copies of programs and data which are available for retrieval by the client computers over the communication link ~~13~~ 14 for use in their processing operations. From time to time, a client computer 11(n) may also store data on ~~the~~ a server computer 12(m), which may be later retrieved by it (the client computer that stored the data) or other client computers for use in their processing operations. In addition, one or more of the server computers 12(m) may, as "compute servers," perform certain processing operations in response to a remote request therefor from a client computer 11(n), and return the results of the processing to the requesting client computer 11(n) for use by them (that is, the requesting client computers 11(n)) in their subsequent processing. In either case, the server computers may be generally similar to the client computers 11(n), including a system unit, video display unit, and operator input devices, and may be usable by an operator for data processing operations in a manner similar to a client computer. Alternatively, at least some of the server computers may include only processing, memory, mass storage, and network interface elements for receiving and processing retrieval, storage, or remote processing requests from the client computers, and

FINNEGAN
HENDERSON
FARABOW
GARRETT &
DUNNER LLP

1300 I Street, NW
Washington, DC 20005
202.408.4000
Fax 202.408.4400
www.finnegan.com

generating responses thereto. It will be appreciated that a client computer 11(n) may also perform operations described herein as being performed by a server computer 12(m), and similarly a server computer 12(m) may also perform operations described herein as being performed by a client computer 11(n).

On page 7, please amend the paragraph starting on line 10 as follows:

An interface provides a mechanism by which a set of methods may be declared. In that connection, an interface identifies each method that is declared by the interface by, for example, a name, identifies the data type(s) of argument(s) that are to be provided for the method, the data type(s) of return values that are to be returned by the method, and identifiers for exceptions which can be thrown during processing of the method. A class may indicate that it implements a particular interface, and in that connection will include the program code which will be used in processing all of the methods which are declared in the interface. In addition, different classes may indicate that they implement the same interface, and each will have program code which will be used in processing all of the methods which are declared in the interface, but the program code provided in each class ~~to~~ for use in processing the methods may differ from the program code provided in the other classes which is used in processing the same methods; thus, an interface provides a mechanism by which a set of methods can be declared without providing an indication of the procedure which will be used in processing any of the methods. An interface may be declared independently of the particular class which implements the method or methods which can be invoked using

FINNEGAN
HENDERSON
FARABOW
GARRETT &
DUNNER LLP

1300 I Street, NW
Washington, DC 20005
202.408.4000
Fax 202.408.4400
www.finnegan.com

the interface. In that regard, a class that invokes the method and a class that actually implements the method will not need to share a common super-class.

On page 8, please amend the paragraph starting on line 11 as follows:

The invention provides an arrangement which facilitates the remote invocation, by a program executing in an execution environment 20 by a client computer 11(n), of methods implemented by classes on a server computer 12(m). In executing a method, the server computer 12(m) will also provide an execution environment 24 for processing, under control of a control module 28, the Java method. In that operation, the Java virtual machine which provides the execution environment ~~21~~ 24 includes a class loader 25 (which may be similar to the class loader 21) that, under control of the control module 28, can dynamically link an instance of the class 26, to enable the method to be processed in the execution environment 24, and instances of other classes (also generally represented by reference numeral 26) which may be needed to process the remotely-invoked method. In that operation, the control module 28 effectively enables the class loader 25 to retrieve an uninstantiated class for the method to be invoked, from a plurality of uninstantiated classes which are generally identified by reference numeral 27, instantiate it (that is, the uninstantiated class which provides the method to be invoked) and link it as a class instance 26 into the execution environment. In addition, the class loader 25 can discard the class instances 26 when processing of the method has terminated. It will be appreciated that, if class instances 26 has been discarded, it may be reloaded by the class loader 25 at a later point if it is then needed.

FINNEGAN
HENDERSON
FARABOW
GARRETT &
DUNNER LLP

1300 I Street, NW
Washington, DC 20005
202.408.4000
Fax 202.408.4400
www.finnegan.com

On page 9, please amend the paragraph starting on line 3 as follows:

To facilitate remote invocation of a method, the control module 19 of the client computer's execution environment ~~21~~ 20 makes use of one or more stub class instances generally identified by reference numeral 30 which are provided as part of the execution environment ~~21~~ 20 in which the various class instances 22, including the class instance which is invoking the remote method, are being processed. Each stub class instance 30 is an instance of an uninstantiated stub class 31, which the server computer 12(m) may maintain for the various class instances 26 and uninstantiated classes 27 which the server computer 12(m) has "exported," that is, which the server computer 12(m) makes available to client computers 11(n) for use in remote invocation of methods provided thereby. An uninstantiated stub class 31 includes declarations for the complete set of interfaces for the particular remote uninstantiated class 27 which implements the remote method to be invoked, and also provides or invokes methods which facilitate accessing of the remote method(s) which are implemented by the remote class. The uninstantiated stub class 31, when it is instantiated and provided to the execution environment 20 of the client computer 11(n) as a stub class instance 30, effectively provides the information which is needed by the control module 19 of the execution environment 20 of the invoking Java program, so that, when a remote method that is implemented by its associated class is invoked by a Java program running in a particular execution environment, the remote method will be processed and the return value(s) provided to the invoking Java program. In one embodiment, the arrangement by which the stub class instance may be provided to the execution environment 20 is similar to that described in the aforementioned Waldo, et al., patent application.

FINNEGAN
HENDERSON
FARABOW
GARRETT &
DUNNER LLP

1300 I Street, NW
Washington, DC 20005
~~202.408.4000~~
Fax 202.408.4400
www.finnegan.com

On page 10, please amend the paragraph starting on line 11 as follows:

Similar operations will be performed if client computer 11(n) and server computer 12(m) are implemented on different physical computers. In that case, in response to a remote invocation, the client computer 11(n) that is processing the invoking class instance 22, under control of the control module 19 for the execution environment ~~10~~ 20 for the invoking class instance 22, will use the appropriate stub class instance 30 to communicate over the network represented by the communication link 14 with the server computer 12(m) which implements the remote method to enable it (that is, the server computer 12(m)) to establish an execution environment 24 for the class which implements the remote method, and to use the class loader 25 to load an instance of the class as a class instance 26. In addition, the client computer 11(n), also using the appropriate stub class instance 30, will provide any required parameter values to the server computer 12(m) over the network 14. Thereafter, the server computer 12(m) will process the remote method using parameter values so provided, to generate result value(s) which are transferred over the network to the client computer 11(n), in particular to the appropriate stub class instance 30. The client computer 11(n) will, after it receives the result value(s) from the network, provide them to the invoking class instance 22 for its processing.

On page 10, please amend the paragraph starting on line 26 as follows:

In any case, when the control module 19 of the client computer's execution environment 20 determines that a reference to the remote object has been received, if it

FINNEGAN
HENDERSON
FARABOW
GARRETT &
DUNNER LLP

1300 I Street, NW
Washington, DC 20005
202.408.4000
Fax 202.408.4400
www.finnegan.com

determines that the stub class instance 30 is not present when it receives the reference, it will attempt to obtain the stub class instance 30 from, for example, the server computer 12(m) which implements the remote method, and enable the stub class instance 30 to be dynamically loaded in the execution environment 20 for the invoking class instance 22. A reference to the remote object may be received, for example, either as a return value of another remote method invocation or as a parameter that is received during another remote method invocation. The stub class instance may be dynamically loaded into the execution environment in a manner similar to that used to load class instances 22 in the execution environment ~~22~~20. The execution environment 20 is provided with a stub class loader 33 which, under control of the control module 19, will attempt to find and load the stub class instances 30 as required by the class instances 22 processed in the execution environment. The location of a particular server computer 12(m) that maintains the class that implements a method to be invoked remotely may be included in the call from the invoking class instance or may be made known to the stub class loader 33 through another mechanism (not shown) maintained by the client computer 11(n).

On page 11, please amend the paragraph starting on line 14 as follows:

However, if the stub class loader 33 is not otherwise notified of which server computer 12(m) maintains the class which implements a method which may be invoked remotely, it may use the nameserver computer 13 to provide that identification. The identification may comprise any identifier which may be used to identify a server computer 12(m) or other resource which is available on the network 14 and to which the

FINNEGAN
HENDERSON
FARABOW
GARRETT &
DUNNER LLP

1300 I Street, NW
Washington, DC 20005
202.408.4000
Fax 202.408.4400
www.finnegan.com

server computer 12(m) can respond. Illustrative identifiers include, for example, a network address which identifies the server computer and/or resource, or, if the network 14 is or includes the Internet, an identifier to, for example, a World Wide Web resource which may provide the identification or a "uniform resource locator" ("URL") which provides a uniform mechanism for identifying resources that are available over the Internet. The server computer 12(m) which implements the remote method, in response to a request from the client computer 11(n) will provide stub class instance 30 which the client computer 11(n) may load into the execution environment ~~21~~ 20 to thereafter enable the remote invocation to be initiated.

On page 13, please amend the paragraph starting on line 10 as follows:

With this background, the operations performed by client computer 11(n), server computer 12(m) and, if necessary, nameserver 13 in connection with obtaining and dynamic loading of a stub class instance when a reference to a remote method is received will be described in connection with the flow chart depicted in FIG. 2. In addition, operations performed by the client computer 11(n) and server computer 12(m) in connection with remote invocation of a method using the stub class instance will be described in connection with the flow chart depicted in FIG. 3. With reference initially to FIG. 2, the execution environment control module 19 will, when it receives a reference to a remote method, ~~will~~ initially determine whether an appropriate stub class instance is present in the execution environment 20 to facilitate invocation of the remote method (step 100). If the control module 19 determines that such a stub class instance 30 for the remote method is present in the execution environment, it may continue other

FINNEGAN
HENDERSON
FARABOW
GARRETT &
DUNNER LLP

1300 I Street, NW
Washington, DC 20005
202.408.4000
Fax 202.408.4400
www.finnegan.com

operations (step 101). However, if the control module 19 determines in step 101 that such a stub class instance is not present in the execution environment 20 for the remote method, the control module 19 will use the stub class loader 33 to attempt to locate and load a stub class instance 30 for the class to process the remote method. In that case, the control module 19 will initially determine whether the invocation from the class instance 22 included a resource locator to identify the server computer 12(m) or other resource which maintains the class for the method to be invoked, or whether it (that is, the control module 19) or the stub class loader 33 otherwise are provided with such a resource locator (step 102). If the control module 19 makes a positive determination in that step, it will sequence to step 103 to enable the stub class loader 33 to initiate communications with identified server computer 12(m) to obtain stub class instance for the class and method to be invoked (step 103). When the stub class loader 33 receives the stub class instance 30 from the server computer 12(m), it will load the stub class instance 30 into execution environment 20 for the class instance ~~21~~ 22 which initiated the remote method invocation call in step 100 (step 104). After the stub class instance 30 for the referenced remote method has been loaded in the execution environment, the method can be invoked as will be described below in connection with FIG. 3.

On page 15, please amend the paragraph starting on line 10 as follows:

After receipt of the result information from the nameserver computer 13, the default stub class instance, under control of the control module 19, will pass result information to the stub class loader 33 (step 113). Thereafter, the stub class loader 33 determines whether the result information from the nameserver computer comprises the

FINNEGAN
HENDERSON
FARABOW
GARRETT &
DUNNER LLP

1300 I Street, NW
Washington, DC 20005
202.408.4000
Fax 202.408.4400
www.finnegan.com

identification for the server computer 12(m) or an indication that no server computer 12(m) is identified as being associated with the class (step 114). If the stub class loader 33 determines that the result information comprises the identification for the server computer 12(m), it (that is, the stub class loader 33) will return to step ~~101~~ 103 to initiate communication with the identified server computer 12(m) to obtain stub class instance for the class and method that may be invoked. On the other hand, if the stub class loader 33 determines in step 114 that the nameserver computer 13 had provided an indication that no server computer 12(m) is identified as being associated with the class and method that may be invoked, the "class not found" exception may be indicated (step 115) and an exception handler called as described above.

On page 15, please amend the paragraph starting on line 22 as follows:

As noted above, the stub class instance 30 retrieved and loaded as described above in connection with FIG. 2 may be used in remote invocation of the method. Operations performed by the client computer 11(n) in connection with remote invocation of the method will be described in connection with the flow chart in FIG. 3. As depicted in FIG. 3, when a class instance 22 invokes a method, the control module 19 may initially verify that a stub class instance 30 is present in the execution environment for remote method to be invoked (step 120). If a positive determination is made in step 120, the stub class instance 30 will be used for the remote invocation, and in the remote invocation will provide parameter values which are to be used in processing the remote method (step 121). Thereafter, the stub class instance 30 for the remote method that may be invoked will be used to initiate communications with the server computer 12(m)

FINNEGAN
HENDERSON
FARABOW
GARRETT &
DUNNER LLP

1300 I Street, NW
Washington, DC 20005
202.408.4000
Fax 202.408.4400
www.finnegan.com

which maintains the class for the remote method (step 122), in the process, the ~~passing~~ parameter values which are to be used in processing the remote method will be passed. It will be appreciated that, if the server computer 12(m) which is to process the method is the same physical computer as the client computer ~~12(n)~~ 11(n) which is invoking the method, the communications can be among execution environments which are being processed within the physical computer. On the other hand, if the server computer 12(m) which is to process the method is a different physical computer from that of the client computer ~~12(n)~~ 11(n) which is invoking the method, the communications will be through the client computer's and server computer's respective network interfaces 15(n) and 16(m) and over the network 14.

On page 16, please amend the paragraph starting on line 13 as follows:

In response to the communications from the stub class instance in step 122, the server computer 12(m), if necessary establishes an execution environment 24 for the class which maintains the method that may be invoked, and the uses the information provided by the skeleton 32 to create a class instance 26 for that class (step 123). Thereafter, the server computer 12(m), under control of the control module 28, will process the method in connection with parameter values that were provided by stub class instance 30 (step 124). After completing processing of the method, the server computer 12(m), also under control of the control module 28, will initiate communications with the client computer's stub class instance 30 to provide result information to the stub class instance (step 125). In a manner similar to that described above in connection with step 102, if the server computer 12(m) which processed the

FINNEGAN
HENDERSON
FARABOW
GARRETT &
DUNNER LLP

1300 I Street, NW
Washington, DC 20005
202.408.4000
Fax 202.408.4400
www.finnegan.com

method is the same physical computer as the client computer ~~12(n)~~ 11(n) which invoked the method, the communications can be among execution environments 24 and 20 which are being processed within the physical computer. On the other hand, if the server computer 12(m) which processed the method is a different physical computer from that of the client computer ~~12(n)~~ 11(n) which is invoking the method, the communications will be through the server computer's and client computer's respective network interfaces 16(m) and 15(n) and over the network 14. After the stub class instance 30 receives the result information from the server computer, it may provide result information to the class instance 22 which initiated the remote method invocation (step 126), and that class instance 22 can continue processing under control of the control module 19.

FINNEGAN
HENDERSON
FARABOW
GARRETT &
DUNNER LLP

1300 I Street, NW
Washington, DC 20005
202.408.4000
Fax 202.408.4400
www.finnegan.com